N-Adic Subdivision Schemes for Local Mesh Deformation

G. Salomon[†], Antoine Leclerq[††], S. Akkouche[‡] and E. Galin[†]

[†]L.I.G.I.M
Université Claude Bernard Lyon 1
69622 Villeurbanne Cedex
[gsalomon|egalin]@ligim.univ-lyon1.fr

[‡]L.I.G.I.M
Ecole Centrale de Lyon
B.P. 163, 69131 Ecully Cedex
samir@ec-lyon.fr

[††]Infogrames
82-84 rue du 1 Mars 1943
69628 Villeurbanne Cedex
aleclercq@fr.infogrames.com

Abstract

This paper presents an interpolating subdivision scheme based on an N-adic decomposition of the parameter space. This approach enables us to locally deform the surface according to a modification of the normals at the vertices of the control mesh.

Experiments show that the N-adic decomposition provides a better control over the deformations, whereas a dyadic decomposition method often produces smoother surfaces.

Keywords: subdivision surfaces, normal control, mesh deformation, N-adic scheme

1. Introduction

Manipulating efficiently and intuitively triangles meshes is a challenging problem for special effects and animation. Mesh models may be either created from scratch in an interactive modeling environment or obtained from digitized models. In general, those models are further edited. For instance, the surface may be smoothed, deformed, and details or extra features may be added. Subdivision surfaces have proved to be an attractive model for creating complex meshes.

Subdivision surfaces have been in the highlights of the computer graphics community for the past few years. One of the greatest advantage of subdivision algorithms is that they produce smooth surfaces from an arbitrary initial control mesh. Moreover, subdivision surfaces may be structured into a hierarchy, enabling the management of level of detail.

A number of subdivision schemes have been proposed so far and may be sorted into two categories. Approximation techniques move all the vertices of the mesh, whereas interpolation techniques keep the vertices of the control mesh unchanged. Approximation techniques have been most extensively studied as they tend to produce better shapes and are more flexible. Those methods do not provide a tight control of the surface, which is often crucial in surface design and character animation. In contrast, interpolation techniques preserve the vertices of the control mesh, but the smoothness of the surface is more difficult to control.

In general, deformations are performed by moving control points. Local multiresolution deformations are achieved by editing the control mesh at different resolution levels[9, 17, 23, 15].

A very attractive modeling feature is the ability to prescribe the normals of a mesh at given vertices. Performing normal control for subdivision surfaces remains an interesting challenge. Several deformations methods using normal control have been proposed, however most rely on approximation schemes[10, 2] or on corner cutting schemes[16].

In this paper, we present a new local deformation method for interpolating subdivision surface schemes. Control is performed by moving the normals of the vertices of the control mesh. In contrast with dyadic schemes, we use a N-adic decomposition of the parameter space to control the deformation of the subdivision. Our work implements a triadic interpolating scheme with normal control. We compare our method with the dyadic decomposition.

2. Previous work

Several authors have proposed techniques for editing and rendering subdivision surfaces efficiently.

Most editing tools focus on the displacement of the vertices of the control mesh and try to optimize the internal representation of the surface with level of details.

Pulli[17] has presented an editing system where the surface is controlled by interactively moving the vertices of the control meshes, whatever the level of subdivision. The algorithm uses the Loop scheme[14] and deals with sharp edges, such as proposed by Hoppe[11].

Zorin et al.[23] have proposed a similar approach based on signal processing techniques. The surface is split into a set of hierarchical meshes that represent the increasing level of detail added by each finer mesh. The process is scalable, it may be performed automatically and enables local modifications on the surface. Although the Loop scheme has been tested, it can be adapted to other schemes.

Mandal et al.[15] have proposed a scheme for dynamic manipulation of the limit surface created with the modified Butterfly scheme[8, 21] using physically based force controls. The control is provided by tracking vertices at various levels of the subdivision. Although accurate, the surface deformation remains global and computationally expensive. Therefore, it is dedicated to simulation rather than interactive animation.

Several other methods deal with the deformation of the surface by modifying the normals at the vertices of the control mesh. Those methods are restricted to non-interpolating approximation schemes however.

Nasri[16] has proposed a simple yet efficient technique for the Doo-Sabin[6] scheme. The first subdivision step aims at creating new vertices that are relocated to adapt to a given normal. Further subdivision steps simply smooth the mesh.

Halstead et al.[10] has improved the Catmull-Clark[3] scheme in order to directly generate the limit surface. Conditions for interpolating normals are given. Unfortunately, using normals to control the limit surface dramatically increases the resolution cost.

Biermann[2] has proposed another method for approximating various schemes such as Loop or Catmull-Clark. This is achieved by adding a correction step after each classical subdivision step. Resulting points from this correction step are oriented relatively to the normal.

Some authors propose other iterative and easily deformable reconstruction methods. Volino[19] uses blended spheres to perform a N-adic tessellation of triangles. Points and normals from the initial mesh are directly involved in its construction.

Vlachos[20] uses a N-adic tessellation to perform his PN Triangles (or N-patches). Primitives used here for reconstruction are polynomials, based on a more current Bézier patch model. Its construction uses points and normals as well. This method has been proposed with a view to being implemented in the graphics hardware.

3. Background

In this section, we present the notations and some results that are useful in the remainder of the paper.

Subdivision schemes are addressed from both a topological and geometrical point of view. The topological aspect is needed to parametrize the mesh, whereas the geometrical aspect purposes the geometric representation in $\mathbb{R}^3$. The subdivision mesh is obtained by elevating vertices in the parametric domain (Figure 1).

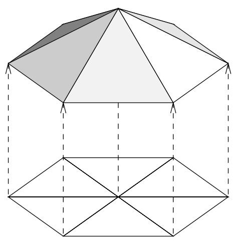

Figure 1: *Elevating vertices of the mesh from parametric domain into space* $\mathbb{R}^3$

The elevation may be split into several transformations associated to the vertices of the mesh. Each transformation itself may be characterized by a mask defining a barycentric combination of neighboring points.

Most subdivision schemes make use of a dyadic decomposition of the parameter space. Edges are split in two by inserting a new vertex in the middle of the edge, hence, triangles are subdivided in four.

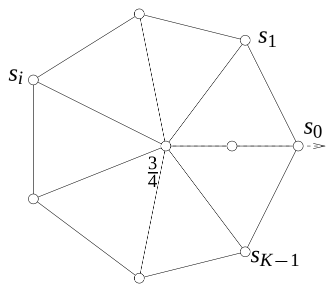

Figure 2: *Modified Butterfly mask for an extraordinary vertex. Vertices involved in the computations belong to the* 1-*neighborhood of the central vertex with valence* K.

Figure 2 illustrates the modified Butterfly[22] mask applied to an extraordinary vertex of the mesh. Coefficients used are as follows, if $K \geq 5$ then :

$$s_i = \frac{1}{K}\left(\frac{1}{4} + \cos\frac{2i\pi}{K} + \frac{1}{2}\cos\frac{4i\pi}{K}\right) \tag{1}$$

If $K = 4$ we have :

$$s_0 = \frac{3}{8}, s_1 = s_3 = 0, s_2 = -\frac{1}{8} \tag{2}$$

Eventually, $K = 3$ yields :

$$s_0 = \frac{5}{12}, s_1 = s_2 = -\frac{1}{12} \tag{3}$$

Let $\mathbf{q}_1$ and $\mathbf{q}_2$ denote the end vertices of an edge. For the modified Butterfly scheme the elevation of the mid-point will be a combination of the two elevations linked to vertices $\mathbf{q}_1$ and $\mathbf{q}_2$ (see Figure 3).

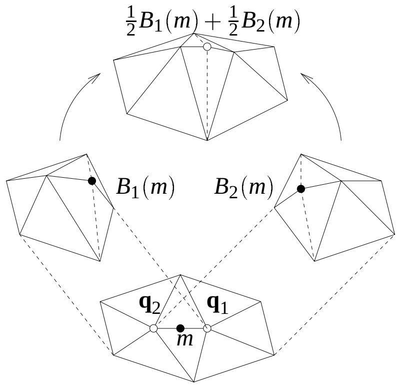

Figure 3: *Inserting a new vertex between two vertices with the modified Butterfly scheme. The final elevation of vertex m is a combination of the elevations at vertices* $\mathbf{q}_1$ *and* $\mathbf{q}_2$.

The coefficients are chosen so that the new vertices inserted in the subdivision should approximate biquadratic splines[6], bicubic splines[18], or quartic[14]. In this paper, we will regard a subdivision surface as the image of a parametrization function. This parametrization function, denoted as B, is defined on a triangle $\Delta\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3$ to be split as follows :

$$\forall m \in \Delta\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3 \begin{cases} B(m) = \sum_{i\in 1,2,3} \alpha_i B_{\mathbf{q}_i}(m) \\ \sum_{i\in 1,2,3} \alpha_i = 1 \end{cases} \tag{4}$$

$B_{\mathbf{q}_i}$ represent the contribution of the vertex $\mathbf{q}_i$ to the elevation of point m. In this paper, we restrict the support of $B_{\mathbf{q}_i}$ to the 1-neighborhood of the vertices $\mathbf{q}_i$. Therefore, only the three functions $B_{\mathbf{q}_1}$, $B_{\mathbf{q}_2}$ and $B_{\mathbf{q}_3}$ participate to the computation of the elevation of a new vertex m inserted in a triangle $\Delta\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3$ as shown in equation 4. In the next section, we address the insertion of new vertices at any location on the triangle $\Delta\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3$.

4. N-adic subdivision scheme

In this section, we describe the subdivision scheme we designed to deform the mesh with normal control. The algorithm relies on the computation of the elevation described in the previous section. Let us recall that the elevation at a given point m is obtained by combining the functions $B_{\mathbf{q}_1}$, $B_{\mathbf{q}_2}$ and $B_{\mathbf{q}_3}$ centered at the 1-neighborhood of the vertices of the triangle $\Delta\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3$. First, we will focus on the computation of the functions $B_{\mathbf{q}_i}$. So as to simplify the presentation, we detail the algorithm that provides a global N-adic scheme for the case $N = 3$. Obviously, the generalization for $N > 3$ is straightforward.

4.1. Computation of a single elevation function

Let $\mathbf{q}$ a vertex of the mesh. We operate on the 1-neighborhood of $\mathbf{q}$, which is defined by a K-sided polygon with vertices $\{\mathbf{q}_0,\cdots,\mathbf{q}_{K-1}\}$. We aim at inserting a new vertex m in the 1-neighborhood of $\mathbf{q}$ and computing its corresponding elevation $B_{\mathbf{q}}(m)$ as a function of existing vertices $\mathbf{q}_0,\cdots,\mathbf{q}_{K-1}$.

To perform this step, we rely on the eigenanalysis developed by Zorin[22] which was first proposed in [1] and [7]. This scheme aims at reproducing polynomials that characterize the local geometry for specific vertices. It may be generalized to any new vertex $m = \rho e^{i\theta}$ over the first triangle. Formulas may be derived for other neighboring triangles by circular permutations. The elevation function $B_{\mathbf{q}}(m)$ may be written as a barycentric combination of the elevations computed at the vertices of the 1-neighborhood of $\mathbf{q}$.

$$B_{\mathbf{q}}(m) = s_{\mathbf{q}} B_{\mathbf{q}}(\mathbf{q}) + \sum_{i=0}^{k-1} s_i^{(\rho,\theta)} B_{\mathbf{q}}(\mathbf{q}_i) \tag{5}$$

The coefficients $s_i^{(\rho,\theta)}$ are functions of ρ and θ, and depend on the connectivity value K. If $K \geq 5$, we have :

$$s_i^{(\rho,\theta)} = \frac{\rho^2}{K} + 2\frac{\rho}{K}\cos\left(\frac{2i\pi}{K}-\theta\right) + 2\frac{\rho^2}{K}\cos\left(\frac{4i\pi}{K}-2\theta\right) \tag{6}$$

If $K = 4$, then :

$$s_i^{(\rho,\theta)} = \frac{\rho^2}{4} + 2\frac{\rho}{4}\cos\left(\frac{i\pi}{2}-\theta\right) + \frac{\rho^2}{4}\cos(i\pi)\cos(2\theta) \tag{7}$$

Eventually, if $K = 3$, then :

$$s_i^{(\rho,\theta)} = \frac{\rho^2}{3} + 2\frac{\rho}{3}\cos\left(\frac{2i\pi}{3}-\theta\right) \tag{8}$$

The coefficient $s_{\mathbf{q}}$ is equal to $1-\rho^2$ whatever K may be. Those coefficients may be compacted into a specific mask parametrized by ρ et θ (see Figure 4).

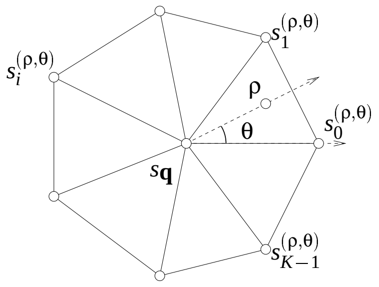

Figure 4: *Subdivision mask for a vertex* $\rho e^{i\theta}$

It is worth noticing that when $\theta = 0$, the new vertex is located on the edge. Therefore, if $\rho = 1/2$, we obtain the modified Butterfly mask for extraordinary vertices[22] when the new vertex is inserted at the middle of the edge. If $\rho = 1/3$, we obtain the coefficient computed by Labsik[13] that characterizes the elevation for the $\sqrt{3}$-subdivision interpolation scheme.

4.2. Triadic subdivision scheme

Creating a N-adic scheme requires that each triangle $\Delta \mathbf{q}_1\mathbf{q}_2\mathbf{q}_3$ should be split into N^2 triangles. Thus, we need to define the elevation for each new vertex m_i as a function of $B_{\mathbf{q}_1}$, $B_{\mathbf{q}_2}$ and $B_{\mathbf{q}_3}$. The final elevation will be defined using the barycentric combination (see equation 4). It is worth noticing that the dyadic case results in a simplified version of the Butterfly scheme.

For each triangle, we need to compute the elevation for two vertices m_1 and m_2 inserted on the radial edge, and for one vertex m_3 at the center of the triangle (see Figure 5). Vertices m_1 et m_2 are given with parameters $\theta = 0$ and $\rho = 1/3$ and $\rho = 2/3$, whereas m_3 is obtained with parameters $\theta = 2\pi/K$ et $\rho = 2/3 \cos \pi/K$. Substituting those parameters into equations (6), (7) and (8) give the corresponding elevation.

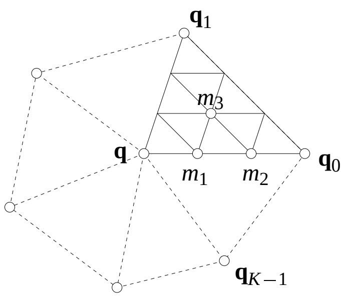

Figure 5: *Creation of a triadic mesh*

In contrast to the well known dyadic schemes, the triadic scheme enables us to add extra constrains in the creation of the vertices of the new mesh at each subdivision step. Those extra constrains will be defined so as to control the deformations in the neighborhood of a vertex of the control mesh (see Figure 6).

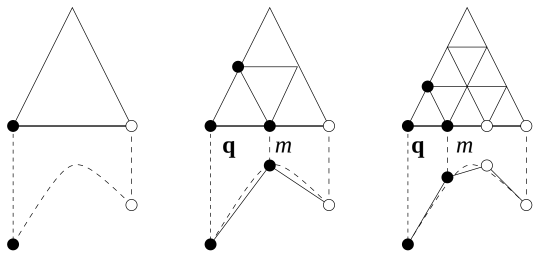

Figure 6: *Comparison of the dyadic (left) and triadic (right) schemes with one subdivision step.*

Figure 7 shows both dyadic and triadic subdivisions on a tetrahedron. Figure 8 illustrates the smoothing of a werewolf like character after several triadic subdivision steps.

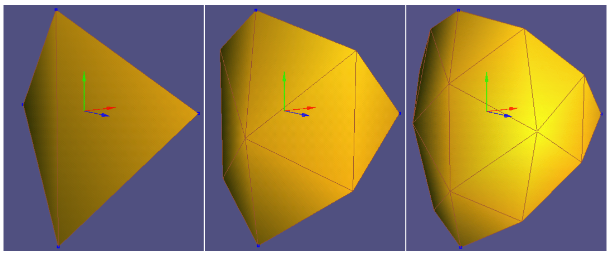

Figure 7: *Structure of a dyadic (middle) and a triadic decomposition (right), the original object is displayed on the left.*

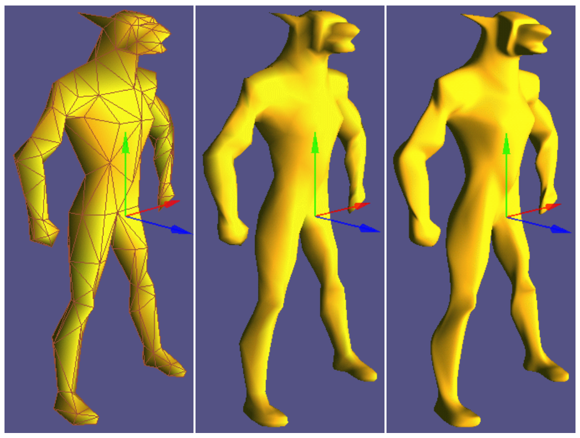

Figure 8: *Recursive subdivision for a triadic scheme. From left to right : original mesh, after one triadic step, after 3 triadic steps.*

Using a triadic subdivision process allows a more accurate rendering of the global shape. This comes from the extra information provided by the extra points added at each triadic step. In general, the triadic scheme provides better control with slightly less smoothing, since it amplifies defaults of reconstruction.

5. Surface deformation

In this section, we present the algorithm used to control the orientation of the subdivision surface in the neighborhood of the vertices of the control mesh. We shall demonstrate the usefulness of the triadic scheme.

The algorithm described in the previous section does not take into account the normal at the vertices of the mesh. We assume that a normal denoted as $N^0_{\mathbf{q}}$ is attached at each vertex $\mathbf{q}$. This normal is often provided with the mesh for rendering purposes. The way this normal has been generated may affect the local deformation of the subdivision surface however. In our implementation, we use the Gauss normals at the vertices of the control mesh.

We deform the surface in the neighborhood of a vertex $\mathbf{q}$ by modifying the orientation of the elevation function $B_{\mathbf{q}}$ associated to this vertex. This is obtained by re-orienting the new normals that result from the subdivision process. The resulting configuration must be aligned with the normal $N_{\mathbf{q}}$ specified by the user. This orientation is characterized by a unique rotation $R_{\mathbf{q}}$ centered at $\mathbf{q}$ transforming $N^0_{\mathbf{q}}$ into $N_{\mathbf{q}}$. $R_{\mathbf{q}}$ is then applied to the points computed by the function $B_{\mathbf{q}}$ (see Figure 9).

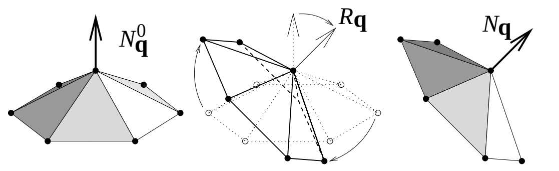

Figure 9: *Local rotation at a vertex $\mathbf{q}$ of the control mesh : after computing $B_{\mathbf{q}}$, each vertex inserted in the 1-neighborhood of $\mathbf{q}$ is rotated using $R_{\mathbf{q}}$ that transforms $N^0_{\mathbf{q}}$ into $N_{\mathbf{q}}$*

The rotation is performed before combining elevations. The modification of the algorithm shows two drawbacks. First, the resulting subdivision surface is only C^0. Extra points inserted can alterate the quality of the final shape at limit of deformed aeras and create creases or cusps (plate 15). This is a slight drawback for us however, as it is still possible to apply extra standard subdivision steps to smooth the surface.

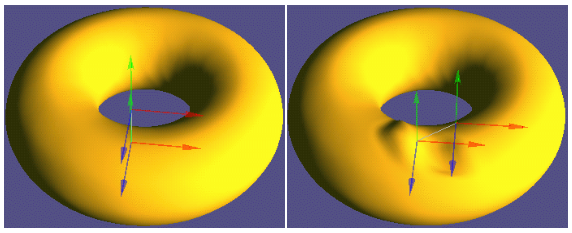

Figure 10: *Perturbing one normal on a torus.*

A more difficult problem is to handle the orientation of the final surface. The algorithm transforms but one elevation function $B_{\mathbf{q}}$ associated with vertex $\mathbf{q}$. Therefore, the computed normal does not match the user-specified normal exactly.

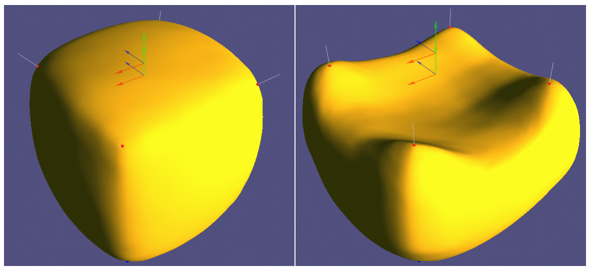

Figure 11: *Deforming one face of a cube.*

This control of the normals allows us to achieve and simple and intuitive control of the shape deformation at the vertices of the control mesh. Deformations are performed after the first subdivision step. The next steps are applied without any deformation, and are used for smoothing only.

Figure 11 shows a cube deformed by perturbing the normals at the four upper corners. Figure 10 illustrates one local deformation on a torus.

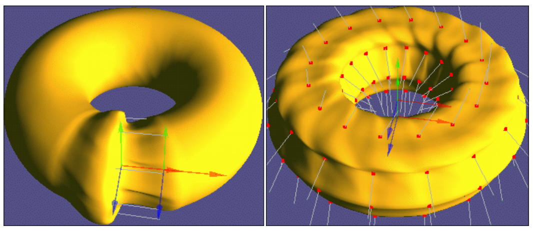

Figure 12: *Several deformations applied on the previous torus.*

Figure 12 shows two examples of deformations. Left image represents a local deformation. Right image shows a model where all normals have been perturbed. The process allows us to complete the design of local curvatures of objects only by moving the normals of its mesh.

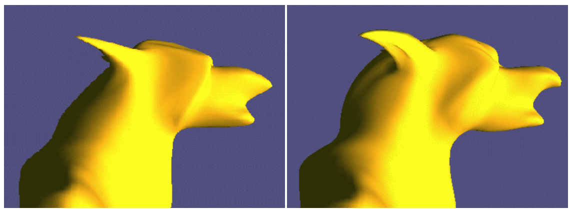

Figure 13: *Redesigning a werewolf character.*

Figure 13 shows the enhanced features on the head of a werewolf character. Left image is obtained without normal

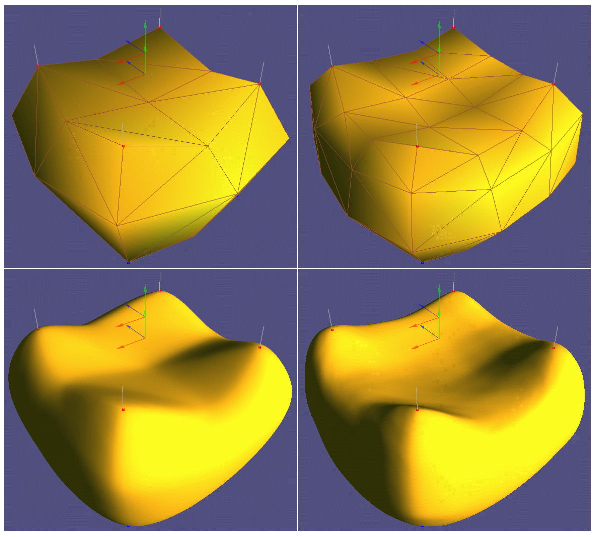

Figure 14: *Deformation of a cube's face for the dyadic and the triadic case.*

deformation, whereas right image has been created by perturbing the control normals. The curved ear has been modeled by turning down the normal at the extremity of the ear. Modifying the normals of the neck gives us a new curvature below the chin and on it's nape. The mouth and the face were modified as well.

Using a triadic subdivision process allows a more accurate rendering of global shape's deformations. This fact comes from the extra points informations introduced at each triadic step.

Figure 14 illustrates the deformations of a cube's face obtained with the dyadic (left) and the triadic (right) scheme. The uppper row shows the structure of the mesh after one deformation step. Bottom row shows the final shaded shape after some smoothing steps. The triadic scheme produces a smoother surface, and creates a deeper carving than the dyadic scheme.

6. Conclusion and future work

We have presented an interpolating subdivision scheme based on an N-adic decomposition of the parameter space. This approach enables us to create an iterative subdivision algorithm.

We have implemented the triadic case so as to locally deform the surface according a modification of the normals at the vertices of the control mesh.

We have compared both dyadic and triadic schemes. The triadic decomposition provides a better control over the deformations, whereas the dyadic decomposition method often produces smoother surfaces.

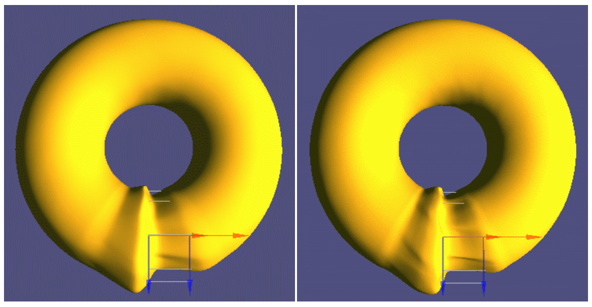

Figure 15: *Deformation of one torus section for dyadic (left) and triadic (right) case. The deformation is quite important here. The local shape obtained by triadic subdivision accurately follows the orientation of normals, but creates creases and cusps. These creases and cusps are not so visible for the dyadic subdivision.*

Several topics need further reasearch. A major drawback of most subdivision methods lies in the recursive form of the algorithms as all subdivision levels need to be computed, even for triangles that will be discarded in the rendering pipeline. The iterative aspect of our method should enable us to avoid subdivision wherever unnecessary. For instance, it could be applied to perform view dependent subdivision.

In our current implementation, the elevation functions $B_{\mathbf{q}}$ are based on a mask operating on the 1-neighborhood of vertex **q**. Studying elevation functions operating on a 2-neighborhood would be interesting as the subdivision surface would probably get smoother.

In the future, we also plan to extend the rotation of the normal at the vertices of the control mesh by using quaternions in order to add some twisting effets.

Our method currently deforms the surface by modifying the position of the vertices of the first level subdivision step. The extra steps are used for smoothing purposes. We plan to extend the deformation to other levels so as to emphasize deformations.

References

1. A.A. Ball and D.J.T. Storry. Analysis of the behaviour of Recursive Subdivision Surfaces Near Extraordinary Points. *ACM Transactions on Graphics.* **7(2)** : 83-102, 1988.

2. H. Biermann, A. Levin and D. Zorin. Piecewise smooth subdivsion surfaces with normal control. *Computer Graphics (Siggraph 2000 proceedings)*. 113-120, July 2000.

3. E. Catmull and J. Clark. Recursively generated B-Spline Surfaces on arbitrary topological meshes. *Computer Aided Design*. **10**(6) : 350-355, 1978.

4. G. Chaikin. An algorithm for high speed curve generation. *Computer Graphics and Image Processing*. **3**(4) : 350-355, 1978.

5. T. DeRose, M. kass, and T. Truong, Subdivision Surfaces in Character Animation. *Computer Graphics (Siggraph 1998 proceedings)*, 85-94, July 1998.

6. D.Doo and M. Sabin. A subdivision algorithm for smoothing down irregularly shaped polyhedrons. *Computer Aided Design (Interactive Techniques proceedings)*. 157-175, 1978.

7. D.Doo and M. Sabin. Analysis of the behaviour of recursive division surfaces near extraordinary points. *Computer Aided Design*, **10**(6) : 356-360, 1978.

8. N. Dyn, D. Levin and J. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transactions on Graphics*, **9**(2) : 160-169, April 1990.

9. D. Forsey and R. Bartels. Hierarchical B-Spline refinement. *Computer Graphics (Siggraph 1988 proceedings)*, **22**(4) : 205-212 August 1988.

10. M. Halstead, M. Kaas and T. DeRose. Efficient, fair interpolation, using Catmull-Clark surfaces. *Computer Graphics (Siggraph 1993 proceedings)*, 35-44, 1993.

11. H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer and W. Stuetzle. Piecewise smooth surface reconstruction. *Computer Graphics (Siggraph 1994 proceedings)*, **28** : 295-302, July 1994.

12. L. Kobbelt. $\sqrt{3}$-Subdivision. *Computer Graphics (Siggraph 2000 proceedings)*, 103-112, July 2000.

13. U. Labsik and G. Greiner. Interpolatory $\sqrt{3}$-Subdivision. *Computer Graphics Forum (Eurographics 2000)*, **19**(3) : 131-138, 2000.

14. C. Loop. Smooth subdivision surfaces based on triangles. *Master thesis, University of Utah, U.S.A*, August 1987.

15. C. Mandal, H. Qin and Baba C. Vermuri. Dynamic Modeling of Butterfly Subdivision Surfaces. *IEEE Transaction On Visualization And Computer graphics*, **6**(3) : 265-286, July-September 2000.

16. A.H. Nasri. Surface interpolation on irregular networks with normal conditions. *Computer Aided Geometrics Design*, **23**(6) : 405-410, 1991.

17. K. Pulli and M. Lounsbery. Hierarchical Editing of Subdivision Surfaces. *Technical Report, University of Washington, U.S.A*, April 1997.

18. M. Sabin The use of piecewise forms for the numerical representation of shape, *PhD thesis, Hungarian Academy of Sciences, Budapest, Hungary*, 1976.

19. P. Volino and N. Magenat Thalmann. The Spherigon : a simple polygon patch for smoothing quickly your polygonal meshes. *Computer Animation 98 proceedings*, 72-79, 1998.

20. A. Vlachos, J. Peters, C. Boyd and J.L. Mitchell. Curved PN Triangles. *ACM Symposium on Interactive 3D Graphics*, 159-166, March 2001.

21. D. Zorin, P. Schröder and W. Sweldens. Interpolating subdivision for meshes with arbitrary topology. *Computer Graphics (Siggraph 1996 proceedings)*, 189-192, 1996.

22. D. Zorin, P. Schröder and W. Sweldens. Interpolating subdivision for meshes with arbitrary topology. *Technical Report, California Institute of Technology, U.S.A*, June 1996.

23. D. Zorin, P. Schröder and W. Sweldens. Interactive Multiresolution Mesh Editing. *Computer Graphics (Siggraph 1997 proceedings)*, 159-168, 1997